Falcon: Live Reconfiguration for Stateful Stream Processing on the Edge

Pritish Mishra

University of Toronto
pritish@cs.toronto.edu

1 Problem Motivation

Emerging mobile and edge applications—such as traffic monitoring, autonomous driving and remote health monitoring—generate significant data requiring low latency of a few tens of milliseconds. In response, cloud providers are adopting a hierarchical model, adding smaller datacenters closer to the network edge [1]. For example, Amazon offers Local Zones, Wavelength, and Outposts, while Microsoft has introduced Azure Stack Edge.

Stream processing is commonly used in these applications, structured as a dataflow graph with vertices representing operators and edges representing data streams along which data tuples propagate between operators. Most operators are stateful, storing contextual information for future computations. For example, recommendation systems leverage users' past activity for personalization, while online model training keeps updated parameters after each round.

Existing stream processing frameworks assume a static operator placement between edge and cloud datacenters. However, the smaller size of edge datacenters leads to higher storage and compute costs, making continuous operation inefficient. Thus, edge stream processing frameworks should adapt to workload changes by seamlessly reconfiguring operator instances and placements. For instance, an additional operator instance should be spawned on an edge datacenter only when bandwidth cost reductions outweigh the higher CPU costs.

Disruption-free reconfiguration of stateful operators poses four key challenges: (i) preserving state correctness during concurrent state migration; (ii) maintaining in-order processing semantics and ensuring tuples are processed exactly once, requiring fault tolerance; (iii) migrating large open windows while preserving windowing semantics; and (iv) addressing *source mobility*, which involves migrating *keys* as mobile data sources move between edges.

2 Our Solution

Our solution, Falcon is the first stream processing framework designed for the hierarchical edge-cloud that enables seamless *stateful* operator reconfiguration and supports source mobility. Falcon uses a novel *live key migration* protocol that combines four techniques:

Dual routing creates a temporary duplicate data flow that routes tuples to both the source and destination instances. This technique allows Falcon to mask the latency of state transfer to the new operator by continuing to run the application on the original instance.

Marker-based synchronization injects special *punctuation marker* tuples into the live data stream to demarcate the phases of the protocol. This avoids the need for lengthy message exchanges to coordinate between source and destination which would incur latency spikes due to network delays.

Lastly, *emission filters* allow an operator instance to synchronize state by processing buffered tuples without emitting output, thus avoiding the need to later deduplicate emitted tuples.

To tie these together, Falcon adopts the *late binding* approach to tuple routing [2], but adapts it to more challenging problems of *stateful* operator reconfiguration and data source mobility.

Falcon also includes a *source mobility protocol* that detects when a source has moved between edges and adapts to this movement by seamlessly migrating the processing and state belonging to this source. This protocol also handles data source "ping-pong" where a source moves back and forth between two edge nodes.

Our evaluations in geo-distributed edge-cloud deployments show that Falcon reduces the length of processing interruptions and their impact on latency by 2 to 4 orders of magnitude compared to the existing state-of-the-art frameworks.

References

[1] Tobias Meuser et al. "Revisiting Edge AI: Opportunities and Challenges". In: *IEEE Internet Computing* 28.4 (2024), pp. 49–59.

[2] Brian Ramprasad et al. "Shepherd: Seamless Stream Processing on the Edge". In: *IEEE/ACM Symposium on Edge Computing (SEC)*. 2022.